

日本国特許庁  
JAPAN PATENT OFFICE

19.11.2004

別紙添付の書類に記載されている事項は下記の出願書類に記載されて  
いる事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed  
with this Office.

出願年月日      2003年11月10日  
Date of Application:

出願番号      特願2003-380114  
Application Number:  
[ST. 10/C] : [JP2003-380114]

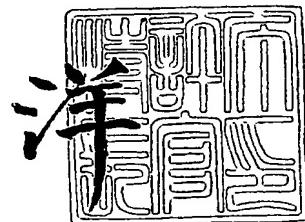
出願人      独立行政法人科学技術振興機構  
Applicant(s):



2005年 1月13日

特許庁長官  
Commissioner,  
Japan Patent Office

小川



【書類名】 特許願  
【整理番号】 Y2003-P218  
【あて先】 特許庁長官殿  
【国際特許分類】 G06F 17/00  
【発明者】  
  【住所又は居所】 盛岡市盛岡駅西通り 1-2-1-802  
  【氏名】 曽我 正和  
【発明者】  
  【住所又は居所】 盛岡市上の橋町 5-5-201  
  【氏名】 猪股 俊光  
【特許出願人】  
  【識別番号】 503360115  
  【氏名又は名称】 独立行政法人科学技術振興機構  
【代理人】  
  【識別番号】 100105371  
  【弁理士】  
  【氏名又は名称】 加古 進  
【手数料の表示】  
  【予納台帳番号】 045414  
  【納付金額】 21,000円  
【提出物件の目録】  
  【物件名】 特許請求の範囲 1  
  【物件名】 明細書 1  
  【物件名】 図面 1  
  【物件名】 要約書 1

**【書類名】特許請求の範囲****【請求項1】**

鍵データを格納した不揮発性メモリで構成される鍵レジスタと、  
該鍵レジスタに格納された鍵データを1ビットづつ参照するために、ビット位置を示す  
鍵カウンタと、

デジタル署名に用いるダイジェスト・データを格納するダイジェスト・レジスタと、  
前記鍵カウンタにより参照された1ビットの鍵データが0のときは前記ダイジェスト・  
レジスタの内容を1、1のときは前記ダイジェスト・レジスタの内容をそのまま出力する  
ゲートとを備え、

一般命令とともに、前記鍵レジスタ、前記鍵カウンタ、及び前記ダイジェスト・レジス  
タを操作して前記ダイジェスト・データからデジタル署名を求めるための複数の署名専用  
命令を有する

ことを特徴とするセキュア・プロセッサ。

**【請求項2】**

請求項1に記載のセキュア・プロセッサにおいて、

プロセッサの走行モードとして、一般モードとセキュリティ・モードを有し、  
前記セキュリティ・モードを表示するセキュリティ・レジスタを備えるとともに、前記  
セキュリティ・モードをセットする一般命令及びリセットする署名専用命令を有し、  
前記一般命令は一般モードのときに有効となり、前記署名専用命令はセキュリティ・モ  
ードのときに有効となることを特徴とするセキュア・プロセッサ。

**【請求項3】**

請求項2に記載のセキュア・プロセッサにおいて、

前記セキュリティ・モード設定命令は、前記セキュリティ・レジスタをセットすると同  
時に、前記鍵カウンタを初期設定し、

前記署名専用命令は、署名計算を鍵レジスタの1ビット分実行する命令の実行と同時に  
鍵カウンタをデクリメントして、順次ビットごとの署名計算が進行し、前記鍵カウンタが  
0のときのみにセキュリティ・モードをリセットすることを特徴とするセキュア・プロセ  
ッサ。

**【請求項4】**

請求項3に記載のセキュア・プロセッサにおいて、

前記ダイジェスト・レジスタに設定されたダイジェスト・データが少なくとも「 $2^n$ 」  
( $n = 1, 2, \dots$ ) ではないことを検出する手段を備え、

前記セキュリティ設定命令は、「 $2^n$ 」ではないことを条件に、前記鍵カウンタを初期  
設定することを特徴とするセキュア・プロセッサ。

**【請求項5】**

請求項3又は4に記載のセキュア・プロセッサにおいて、

セキュア・プロセッサには、主メモリに接続しており、

前記署名専用命令は、前記デジタル署名を求める演算の演算結果を、前記主メモリの特  
定の領域のみに格納して、デジタル署名演算の最終結果を前の演算結果に上書きすること  
を特徴とするセキュア・プロセッサ。

**【請求項6】**

請求項1～5のいずれかに記載のセキュア・プロセッサを組み込んだICカード。

**【書類名】**明細書

**【発明の名称】**セキュア・プロセッサ

**【技術分野】**

**【0001】**

本発明は、汎用マイクロプロセッサのアーキテクチャ（論理構成）に関し、特に、デジタル署名等のセキュリティ技術にも用いられるマイクロプロセッサのアーキテクチャに関する。

**【背景技術】**

**【0002】**

コンピュータ誕生以来約60年の間、プロセッサの機能・性能の向上は100万倍のオーダーにも達する。改善の主たる担い手は素子・回路の機能・性能の改善による。その次の副たる担い手は、アーキテクチャの改善である。そのアーキテクチャ面での改善も殆どが性能向上に向けられてきた。最近20年間程度ではごく一部で、信頼性向上や電力低減のためのアーキテクチャ改善もあった。しかしながら、セキュリティ改善のために行われたアーキテクチャ改善の例はない。

ただし、セキュリティ改善の目的のために公開鍵暗号計算（例えばRSA暗号計算）専用のプロセッサが使用されている例はある。それはIBM、富士通、松下通信、NTTデータ等のセキュリティ専用プロセッサである。これは署名演算や暗号計算だけを引き受ける専用の付加プロセッサである。別に汎用の主プロセッサが存在することが前提である。これらは、もともと専用プロセッサ全体が署名演算・暗号計算だけしか受け付けない機能限定プロセッサであるから、鍵データを他目的に使われる危険も容易に避けられる。

なお、RSAによる暗号計算のアルゴリズムについては、例えば、非特許文献1に示した文献を参考にされたい。

**【非特許文献1】**Cetin Kaya Koc "High-Speed RSA Implementation Vresion 2.0" RSA Data Security, Inc. 1994(<ftp://ftp.rsasecurity.com/pub/pdfs/tr201.pdf>)

**【発明の開示】**

**【発明が解決しようとする課題】**

**【0003】**

本発明の目的は、汎用機能を持ち、なおかつ、セキュリティ機能（すなわち鍵データの安全保管とデジタル署名計算の高速化）も持つプロセッサの提供である。

**【課題を解決するための手段】**

**【0004】**

上述の目的を達成するために、本発明は、鍵データを格納した不揮発性メモリで構成される鍵レジスタと、該鍵レジスタに格納された鍵データを1ビットづつ参照するために、ビット位置を示す鍵カウンタと、デジタル署名に用いるダイジェスト・データを格納するダイジェスト・レジスタと、前記鍵カウンタにより参照された1ビットの鍵データが0のときは前記ダイジェスト・レジスタの内容を1、1のときは前記ダイジェスト・レジスタの内容をそのまま出力するゲートとを備え、一般命令とともに、前記鍵レジスタ、前記鍵カウンタ、及び前記ダイジェスト・レジスタを操作して前記ダイジェスト・データからデジタル署名を求めるための複数の署名専用命令を有することを特徴とするセキュア・プロセッサである。

これにより、汎用機能を持ち、不揮発性メモリの鍵レジスタに格納した鍵データを直接的には読めないため、セキュリティ機能も持つプロセッサを提供することができる。

このプロセッサの走行モードとして、一般モードとセキュリティ・モードを有し、前記セキュリティ・モードを表示するセキュリティ・レジスタを備えるとともに、前記セキュリティ・モードをセットする一般命令及びリセットする署名専用命令を有し、前記一般命令は一般モードのときに有効となり、前記署名専用命令はセキュリティ・モードのときに有効となるとすることもできる。

**【0005】**

前記セキュリティ・モード設定命令は、前記セキュリティ・レジスタをセットすると同

時に、前記鍵カウンタを初期設定し、前記署名専用命令は、署名計算を鍵レジスタの1ビット分実行する命令の実行と同時に鍵カウンタをデクリメントして、順次ビットごとの署名計算が進行し、前記鍵カウンタが0のときのみにセキュリティ・モードをリセットする構成とし、一旦、デジタル署名の演算過程に入ると、終了するまで（鍵カウンタが0となるまで）、演算過程から抜けられず、演算過程の中間結果からプログラムにより鍵データを推定することを不可能としている。

前記ダイジェスト・レジスタに設定されたダイジェスト・データが少なくとも「 $2^n$ 」( $n = 1, 2, \dots$ )ではないことを検出する手段を備え、前記セキュリティ設定命令は、「 $2^n$ 」ではないことを条件に、前記鍵カウンタを初期設定して、鍵データを間接的に読み出すことができないようにしてもよい。

セキュア・プロセッサには、主メモリに接続しており、前記署名専用命令は、前記デジタル署名を求める演算の演算結果を、前記主メモリの特定の領域のみに格納して、デジタル署名演算の最終結果を前の演算結果に上書きする構成として、中間結果を主メモリに残すことを不可能としてもよい。

上述のセキュア・プロセッサを組み込んだICカードでは、署名演算をICカード内で実行でき、鍵データをICカード外部に取り出す必要がなく、署名演算を安全に実行することができる。また、鍵データをICカードからプログラムにより外部に取り出すことは不可能である。

#### 【発明の効果】

##### 【0006】

本発明のセキュア・プロセッサは、汎用性とセキュリティ機能の二律背反性を新しいアーキテクチャを考案することによって解決している。

上述のセキュア・プロセッサが用いられている個人認証用のICカードの主たる機能は署名演算である。本発明のセキュア・プロセッサを用いることにより、セキュリティ機能で署名演算を行うばかりでなく、キャッシュカード機能、クレジットカード機能、改竄防止機能、通行料金支払い機能、予約チケット機能、等々の応用に対して、異なる汎用の応用プログラムをプロセッサの上で走行することにより、対応することができる。

#### 【発明を実施するための最良の形態】

##### 【0007】

個人認証において、セキュリティ機能を突き詰めていくと、個人鍵データを署名演算にのみ使えるようにし、他のあらゆる操作を拒否することに行き着く。これを実現するために、本発明のセキュア・プロセッサは大きく分類して2つの構成を有している。

A：個人鍵データが署名演算等において使われるときは、上位から順番に1ビットづつ参照される動作のみで、他の動作はない。このことに着目して、個人鍵を主メモリや汎用レジスタではなく、独自の不揮発性専用レジスタに格納する。そして、その専用レジスタには、全データを外部から読み取るパスはなくする。上位から順に1ビットづつ参照する演算のためのパスのみを作る。

B：セキュア・プロセッサにおいて、従来のプロセッサには存在しなかったセキュリティ・モードと称するモードを導入する。これはプログラム走行の環境を制約する条件でもある。セキュリティ・モードにおいては通常の命令語は命令として解読されない。署名演算でのみ使用される特殊な命令語のみが命令として機能する。

これらの手法によって、本発明のセキュア・プロセッサは、如何なるプログラムによつても個人鍵データを外部へ漏洩せしめることは不可能となる。

以下に、図面を用いて、上述のA, Bの両機能を備えたセキュア・プロセッサの構成例を詳しく説明する。

##### 【0008】

#### <プロセッサの構成>

図1は、セキュア・プロセッサの実施形態の1例を示すブロック図である。図2は、セキュア・プロセッサ内の鍵レジスタ部分のブロック図である。図3は、実施形態のセキュ

ア・プロセッサの命令フォーマットである。図4は、セキュア・プロセッサのオペコード（OPコード：命令コード）例である。

### 【0009】

#### <セキュア・プロセッサの内部構成>

図1に示したプロセッサ100は、署名演算のために、なるべく長い語長（64ビット）を有し、16倍長語までの計算が1個の命令できる。また、主メモリ200とのアドレス単位も語単位（64ビット単位）である。主メモリの容量は、 $64\text{ビット} \times 64\text{K語} (512\text{KB} = 4\text{Mbits})$  である。これらについては、後で詳しく説明する。

なお、以下に説明するセキュア・プロセッサの語単位として、64ビットを用いた構成としているが、これは、鍵データとの署名演算から、なるべく長い語長を選択したためであり、演算速度等やハードウェア量との関係から、決定すればよい事項である。

### 【0010】

#### (セキュア・プロセッサ内部構成の説明)

図1のCPU100内には、命令レジスタ（IS：Instruction Register）143、メモリデータ・レジスタ（MD：Memory Data Register）144、プログラム・カウンタ（PC：Program Counter）145、F-オペランド・カウンタ（FC：F-operand Counter）146、T-オペランド・カウンタ（TC：T-operand Counter）147が用意されている。これらは、主メモリ200との間にあり、主メモリ200から、命令やデータを読み出したり、書き込んだりするためのレジスタである。

プログラム・カウンタ（PC）145の内容により、主メモリ200から命令レジスタ（IS）143に、図3（a）に示すフォーマット（64ビット）の命令が読み出される。この命令により、図3（a）のMFやMTのフィールドで指定されたアドレスモード（図3（b）参照）に従って、F-オペランド・カウンタ146、T-オペランド・カウンタ147にアドレス等が設定され、主メモリ200からメモリデータ・レジスタ（MD）144にデータが読み出されたり、メモリデータ・レジスタ144（MD）内のデータが主メモリ200に書き込まれたりする。

### 【0011】

倍長演算命令では、図3のLフィールドに指定された語長分、F-オペランド・カウンタ146やT-オペランド・カウンタ147は、インクリメント（1づつ増加）されて、オペランドが主メモリ200から次々に読み出されるか書き込まれることで、16倍長語までの処理が1個の命令できる。

汎用演算器（ALU：Arithmetic and Logic Unit）164は、一般的な演算（加算、減算、論理演算等）を行う演算器であり、乗算器（MPY：MultiPLY unit）162は、 $64\text{ビット} \times 64\text{ビット}$  の掛け算を行う演算器である。

PFレジスタ（Program Status Flag）148は、命令の実行によりセットされる4ビットのフラグであり、図4-1（a）、図4-2（b）に示されているオペコード表のPSWの項に示されているものがセットされる。オペコード表のPSW（Program Status Word）のN、Z、V、Cは、それぞれ、負（Negative）、ゼロ（Zero）、オーバーフロー（Overflow）、キャリ（Carry）を示している。

演算用レジスタ（R0～RF）110は、一般用の演算命令等で用いられる汎用のレジスタである。この汎用レジスタを用いる演算命令は、図3（a）のMF又はMTに図3（b）のレジスタ直接を指定して、F又はTオペランドのフィールドにレジスタ番号を指定する。バッファ・レジスタ（B0、B1）141、142は、演算途中の結果等を格納しておくレジスタである。

上述の命令の動作や各レジスタの機能は、詳しくは、図3に示した命令のフォーマットと図4-1、図4-2のオペコード表（命令コード表）等を参照されたい。図4-1（a）及び図4-2（b）はオペコード表であり、図4-2（c）は、オペコード表のフィールドごとに使用されている記号の説明であり、図4-2（d）は、オペコード表の動作の項の表記の説明であり、図4-2（e）は、図4-2（c）、（d）の\*で示しているとの補足説明である。

オペコード表のOP, SOP, MF, MT, L, F, T, Sの各項は、図3(a) (b) の命令フォーマットの説明と図4-2(c)の記号の説明を参照されたい。また、ニモニックは、命令の略号であり、今後、各命令を参照するときは、このニモニックを使用している。各命令の動作は、動作の項に記載されている。属性は、命令の分類で、例えば、SFTはシフト命令、ADDは加算命令、SUBは減算命令、BITはビット処理命令、MOVは移動命令、JMPはジャンプ命令、LINKはサブルーティンコール命令、BRはPSWによるブランチ命令を示している。なお、SVC命令は、スーパバイザ・コール命令で、図1に示していないITフラグをセットしている。RETはスーパバイザ・コール命令のリターン命令で、ITフラグをリセットしている。

なお、上述のオペコード表の中に、以下で説明する署名演算用の命令も含まれている。これらの命令については、後で説明する。

#### 【0012】

以上説明した構成は、通常の汎用マイクロ・プロセッサの構成と同様のものである。これらは、このセキュア・プロセッサが応用される分野に応じて、変更することができる。例えば、64ビット/語の構成を生かすために、以下の署名演算で用意されているmod演算に有効な命令を、暗号計算用に一般用の命令として用意してもよい。

#### 【0013】

##### <署名演算に関する構成>

以下に説明する構成は、セキュア・プロセッサの署名演算に密着している構成例である。

#### 【0014】

##### (鍵データ漏洩防止)

さて、セキュリティ機能の一つの目的は、鍵データK漏洩の防止である。図1において、鍵データは、不揮発性メモリ（例えばROM）で構成された鍵レジスタ（K0～KF）に格納されている。鍵レジスタ130には、各ユーザの例えR:Key register) 130に格納されている。鍵レジスタ130には、各ユーザの例えR:Key register) 130に格納されている。鍵SAの秘密鍵データ（1024ビット）が書き込まれている。この不揮発性メモリへの鍵SAの書き込みは、例え外部から専用のライタにより書き込むことで行っても良い。

RSA公開鍵方式でのデジタル署名演算の方式では、鍵Kの値は1ビットづつ上位ビットから参照して、別の乗算へ反映させる必要があり、これが唯一の使用方法である。従って、この目的のために、図2のような、鍵レジスタ130から1ビットづつ選択して参照する鍵参照回路を設けている。なお、これらの回路を用いたデジタル署名演算のアルゴリズムについては、後で詳しく説明する。

#### 【0015】

図2の鍵参照回路において、鍵ビット参照カウンタ（KC:Key-bit reference Counter）152は1023から0まで、順次デクリメントする。この鍵ビット参照カウンタ152の内容により、ビット指定ゲート154から、鍵レジスタ130に格納されている鍵データが順次1ビットづつ指定されて参照される。図2からも分かるように、鍵Kを格納する鍵レジスタ130から他ヘデータを転送する語データ並列伝送経路を設けていない。図1に示すように、ビット指定ゲート154からの参照出力は、ダイジェスト・レジスタ120の出力選択ゲート156にのみ使用される。このように、図1、図2に示されているハードウェア構造からして、鍵データを生の形で直接外部へ出すことは不可能である。

#### 【0016】

図1のダイジェスト・レジスタ（D0～D2:Digest register）120は、64ビット×3のレジスタであり、デジタル署名の処理のために、デジタル署名を付加するための本文から抽出されたダイジェスト・データ（特徴データ）を格納するためのレジスタである。

デジタル署名の処理を行う前に、本文の3語長分のダイジェストを作成し、主メモリ200に格納しておく。ダイジェストは、本文のビット構成如何にかかわらず160ビット程度まで圧縮攪拌され（たとえばハッシュ関数SHA-1によって）ランダムなビット構成となっている。しかしながら、このダイジェスト・データを故意に2のような単純な値に設

定することもできる。すると、署名計算結果から鍵Kの値を逆算される危険が発生する。これを防止するためにダイジェスト・レジスタの内容が「 $2^n$ 」( $n = 1, 2, \dots$ )のよ うな単純な値ではないことを検知する必要がある。セキュア・プロセッサでは、単純な値ではないことを「有効パターン」と呼んで署名計算開始の条件としている。図4-1のオペコード表で、DMVと表記されている命令により、Fフィールドで指定されているメモリ200のアドレスに格納されている3語長分のダイジェストをダイジェスト・レジスタ(Dレジスタ)120に格納する。

D レジスタ 120 には、ビットパターン検知ゲート（図示せず）が付属している。ビットパターン検知ゲートは、D レジスタ 120 の全 160 ビットを 16 ビットづつ 10 ブロックに区分し、すべてのブロックが少なくとも 1 個の” 1 ”を持つかを検知している。これにより、格納したデータが「有効パターン」であるかを検出している。

上述した鍵参照回路で読み出された1ビットの鍵データは、Dレジスタ120から読み出されたダイジェスト・データDに対して、D判定ゲート156により反映している。これについては、後述する署名演算で説明する。

[0017]

(ヤキュリティ・モード)

(セキュリティ) さて、上述した鍵レジスタ 130 や鍵参照回路のハードウェア構造からして鍵データ K を生の形で外部へ出すことは不可能なことは自明である。残る問題は上記の計算の中で間接的に使用される鍵データのビット毎の値を計測され、集められるかどうかである。観測対象が複雑な署名処理の最終結果（全 1024 ビット）ならば、それはデジタル署名データであるから鍵データの推測は実用上不可能であることは知られている。そこで各個別に 1 ビット毎にモンゴメリー乗算結果から K の値（0 または 1）を推測されないようにするための他の防護策を次に説明する。

[0018]

署名演算を他の普通の計算と区別し、署名演算の途中で使用される命令を署名以外の目的に悪用されないように防護するために、セキュア・プロセッサでは、プログラム走行モードを普通モードとセキュリティ・モードとに区分している。どちらのモードで走行中かをセキュリティフラグ・レジスタ (SFレジスタ: Security Flag register) 149で示している。

図1のSFレジスタ149にはSF3, SF2, SF1, SF0の4ビットがあるが、  
当面使うのはSF0のみである。

(1) SF0 = 0 : 普通モード

専用プログラム部分、パソコン交信部分、圧縮計算（ダイジェスト・データを得る）

部分

(2) S F 0 = 1 : 署名モード

署名演算実行中（後で説明する図5のフローチャート部分）

署名演算実行中（後で説明する）。

命令セットの中で、一般的命令は、セキュリティ・フラグ SF0 が 0 でないと有効に働く。命令セットの一部に署名演算の途中でのみ使用される命令を用意する。これらのかない。命令 SF0 のマッチングがとれないときは命令は SF0 = 1 でないと有効に働くかない。なお、SF0 のオペコード表で、「無動作 NOP 命令と同じになる。図 4-1 (a), 図 4-2 (b) のオペコード表で、「動作」の欄に [SF = 1] と表示がある命令がこの命令に該当し、SIE, KCJ, AD O, SCMP, SSB, MLS, MDK, MLD, MLL, MLH, MLP の各命令である。なお、MLS, MDK, MLD, MLL, MLH, MLP の命令は、演算結果を格納するための先頭アドレスを指定する S フィールドを有しており、他の命令とフォーマットが異なっている。

[0019]

SF0=1を設定する命令(図4-1(a):SIG)は、必ず同時に、KC=03F  
 F (=1023d), 0000~000F番地の1024ビットデータ=0000…0  
 001に設定する。SIG命令は、ダイジェスト・レジスタ120の内容が有効パターン  
 001に設定する。これは、ダイジェスト・レジスタ120に”2”と設定されて  
 ないと有効に働かない。

、 $2^K \bmod N$ から鍵Kを逆算されることを防ぐ。

【0020】

署名演算の途中で使用する計算結果の格納箇所は、以下の主メモリの上位64番地の固定番地を使用している。

- (1) 0000~000F 16番地 (1024ビットデータ)
- (2) 0010~001F 16番地 (1024ビットデータ)
- (3) 0020~002F 16番地 (1024ビットデータ)
- (4) 0030~003F 16番地 (1024ビットデータ)

計算結果が1024ビットの場合は(1) 0000~000F番地の16番地へ格納される。計算結果が2048ビットの場合は(1) (2) 0000~001Fの32番地へ格納される。計算結果を一旦退避させたいときは、(3) (4) 0020~003Fへ退格納される。計算結果を一旦退避させたいときは、(3) (4) 0020~003Fへ退格納される。計算結果を一旦退避させたいときは、(3) (4) 0020~003Fへ退格納される。なお、SF0=1が解除されてSF0=0になったあとでは、避せることのみできる。なお、SF0=1が解除されてSF0=0になったあとでは、普通のMOV命令(図4-1(a)参照)等で上述の上位固定番地の内容を他番地へ移せる。

【0021】

そして、署名演算が終了するまで、即ち、鍵カウンタKCがゼロとなるまで、SF0をリセットする命令(SIE)を動作させることができない(図4-1(a)参照)。これについては後で詳しく説明する。

この結果、ビット毎の計算結果は固定の番地のみへどんどん集積し、ビットごとの途中結果は外部へ取り出すことはできず、最後の全ビット積算結果のみ外部へ取り出せる。

【0022】

(署名演算)

RSA公開鍵システムにおけるデジタル署名は、 $D^K \bmod N$  (D: ダイジェストデータ, K: 鍵, N: 特定の整数) を計算することである。

$D^K \bmod N$ を一個の巨大な特殊命令で扱うことは、セキュリティ上は望ましいものの、ハードウェア資源上得策でないので、通常の命令と同程度の大きさの特殊命令を複数個用意して実行する。すると、それらの特殊命令の使い方を変えて、鍵Kを外部へ漏洩する悪意のプログラムを構成できる可能性も発生する。これを防護するのが上述したセキュリティ機能である。このセキュリティ機能を利用する、 $D^K \bmod N$ の計算をおこなう手順を、図5のフローチャートを用いて説明する。図5のフローチャートは、バイナリ法(Binary Method)のアルゴリズムで処理する場合であり、詳しくは、例えば上記の非特許文献1(2.3 The Binary Method (p.10~p.11))を参照されたい。なお、手順中のAはメモリ0000~000F番地の内容である。

【0023】

図5のフローチャートにおいて、初期設定(S312)に必要な、 $1 \rightarrow SF0$ ,  $1 \rightarrow A$ ,  $1023 \rightarrow KC$ の3つの初期動作を1個の命令(SIG命令)で行っている。そして、署名演算のサブルーティンに入る。

ただし、SFをセットするためには、ダイジェスト・レジスタ120の内容が有効パターンであることが条件となる。有効パターンとは、160ビットあるダイジェスト・レジスター内のデータが十分に攪拌され、圧縮されたデータであることを言う。16ビット×120の16ビット毎に少なくとも1個は1があることをハードウェア的に検知する。SF0=1となったあとは、ダイジェスト・レジスタ120内のデータ変更はできない。

【0024】

さて、初期化のあと、 $A^2 \bmod N$ を計算する(S314)。最初は、A=1である。つぎに、 $A \times D \bmod N$ を計算する(S316)。Dは、以下のようにKC(鍵K中の鍵カウンタ(KC)152で指示された位置のビット)の値によって2通りの値をとる。

## 【数1】

```

if Kc=1      then D=D
if Kc=0      then D=1

```

Dは、ダイジェスト・レジスタ D<sub>120</sub>から読み出されたダイジェスト・データ D と、鍵レジスタ 130 から鍵カウンタ (KC) 152 により読み出された Kc とから、ハードウェア・ゲート 156 で作られる。図 4-2 (b) のオペコード表では、MDK 命令である。

、鍵カウンタのデクレメントと上述の Dを得ることを同時に実行する命令である。  
問題の鍵 K は、直接には表面に現れないが、間接的に  $A \times \underline{D}^K \bmod N \rightarrow A$  の D に影響を与える。外部に絶対に漏洩させてはならない K が、間接的に関与する部分は  $A \times \underline{D}^K \bmod N$  の計算部分である。

## 【0025】

これを鍵 K の長さ分、即ち、鍵カウンタ KC 152 が零となるまで (S318 で YES ) 、鍵カウンタ KC 152 をデクレメントして (S320) 、2つの mod 計算 (S31 4. S316) を行う。鍵カウンタ KC 152 が零となると、SF0 を零にリセットして、この署名演算のサブルーティンから抜けることができる (SIE 命令) 。

特定の領域に格納される計算途中の演算結果は、次々に上書きされ、署名演算が終了すると、最終結果として署名されたデータが格納されている。

## 【0026】

上述の mod 計算 (S314. S316) の部分は、乗算命令を中心とする複数ステップがループする構造になっている。mod N の演算は、通常ならば割り算となるが、モンゴメリー乗算と呼ばれる計算手順を使って多数の乗算と1回の減算で実行する。モンゴメリー乗算のアルゴリズムについては、モンゴメリー乗算については、非特許文献 1 (3.8 Montgomery's Method p.46~p.47) を参照されたい。

## 【0027】

(モンゴメリー乗算の手順)

$\underline{A}\underline{D}^K \bmod N$  をモンゴメリー乗算で求めると、次のような式となる (式中の減算 N は引けないときはそのまま) 。

## 【数2】

$$\left[ \frac{\underline{A}R + (\underline{A}R * \underline{D}) \bmod N - \underline{D}}{R} + \frac{(\underline{A}R * \underline{D}) \bmod N * R - \underline{A}DN * \bmod R N}{R^2} \right] - N$$

上記の式に現れる R, R\*, N\* は、いずれも N が与えられた最初の設定時に同時に導出できる定数である。実用上の RSA 公開鍵システムでは、公開パラメータ N は 1024 ビット長に固定されているので、R, R\*, N\* は下記となる。

$$\cdot R = 2^{1024} = 100000\cdots000$$

(R のデータビット長は 1025 ビットになる。)

$$\cdot R^* = R^2 \bmod N = 2^{2048} \bmod N$$

(mod N をとるから、R\* のデータビット長は 1024 ビット以下である。)

$$\cdot N^* = \gamma R - 1 を満足するように N* を計算する。 \gamma は任意の整数である。$$

(N\* のデータビット長は 1024 かまたはそれ以下となる。)

式の形の上では R による除算が 3箇所、mod R が 3箇所存在するが、R の値が特殊な形の  $2^{1024}$  なのでビット操作で済ませられる。

## 【0028】

上述の式を求めるための手順は、網掛け部分が同じであるので、以下の通りである。

(01) AR\* を求める。

乗算、結果は Max. 2048 ビットである。

(02) AR\* N\* を求める。2048 ビットでもオーバーフローするので、AR\*

$m o d R$  を先に求める（下位 L ビットの抽出）。上半分の 1024 ビットを捨てる。

(03) (02)  $\times N^*$  を求める。乗算結果は最大 2048 ビットである。

(04) (03)  $m o d R$  を求める（下位 L ビットの抽出）。上半分の 1024 ビットを捨てる。

(05) (04)  $\times N$  を求める。乗算結果は最大 2048 ビットである。

(06) (01) + (05) を求める。加算結果は最大 2049 ビットである。

(07) (06)  $/ R$  を求める（下位 L ビットの 0 の除去）。下半分の 1024 ビットを捨てる。

(08) (07)  $- N$  を求める。これを X とする。減算は、正負判定して選択して行う。

(09) (08)  $\times D$  を求める（ここで Kc が影響する）。乗算の結果は、 $1024 + 160 = 1184$  ビットである。

(10) (09)  $N^*$  を求めると 2048 ビットでもオーバーフローするので、(09)  $m o d R$  を先に求める（下位 L ビットの抽出）。下半分の 1024 ビットを取り出す。

(11) (10)  $\times N^*$  を求める。乗算結果は最大 2048 ビットである。

(12) (11)  $m o d R$  を求める（下位 L ビットの抽出）。下半分の 1024 ビットを取り出す。

(13) (12)  $\times N$  を求める。乗算結果は最大 2048 ビットである。

(14) (09) + (13) を求める。加算結果は最大 2049 ビットである。

(15) (14)  $/ R$  を求める（下位 L ビットの除去）。下半分の 1024 ビットを捨てる。

(16) (15)  $- N$  を求める。減算は、正負判定して選択する。

なお、上述の (08) および (16) 「正負判定して選択する」としているのは、減算結果が正ならばそのままの値を答えとし、減算結果が負ならば、その値を捨てて、減算する前の値を答えとすることを意味する。

### 【0029】

上述の式を、図 4-1 (a), 図 4-2 (b) に示したオペコード表の命令で行う場合を、図 6 (a) に示す。図 6 (a) に表れた記号の意味は図 6 (b) に示されている。

図 6 (a) で示されているように、上述の手順は、全て、図 4-1 および図 4-2 に示されている、セキュリティ・モードであるときに動作する命令で、計算することができる。

### 【0030】

#### < IC カードへの応用 >

セキュア・プロセッサを IC カードに応用した場合について、図 7 を用いて説明する。さて、デジタル署名は、上述したように、対象メッセージから作成したダイジェストデータに対して、個人鍵で暗号化することで生成される。図 7 (a) は現在用いられている認証 IC カード 310 を示している。現在用いられている認証用 IC カード 310 は、認証 IC カード 310 に個人の鍵データが書き込まれている。メッセージ送り主は、パソコン IC カード 310 に個人の鍵データが書き込まれている。メッセージ送り主は、パソコン 320 に付属するカードリーダ (図示せず) へ乗せ、パソコン 320 の表示画面上の署名 320 が表示される。すると、認証用 IC カード 310 から個人動作を示すボタン等をクリックして起動する。すると、認証用 IC カード 310 から個人の鍵データが読み出され、パソコン 320 内でデジタル署名が生成され、メッセージ本体とペアになってインターネット 330 経由で相手先へ送られる。署名演算がすむと IC カードをリーダから取り除くことで署名動作全体が終了する。

この動作において、署名演算をどこで実施するかでセキュリティレベルが違ってくる。図 7 (a) に示す現在の方法では、個人鍵データがパソコンの中に移るわずかな時間の間にも、盗聴されたりコピーされたりする危険が有りうる。

図 7 (b) に示すセキュア・プロセッサを組み込んだ IC カード 315 の場合は、署名演算能力を有するので、IC カード 315 に対象メッセージを取り込み、IC カード 315 内で署名演算を実施する。IC カード 315 に取り込むデータは、ダイジェスト・データでもよい。IC カード 315 からパソコン 320 へ送られて出てくるのは、デジタル

署名結果であり、鍵データではない。ICカード内で、署名演算を行っているために、鍵データを外部に読み出す必要はなく、上記の危険性はない。署名結果から鍵を逆算することは天文学的時間をする。

しかも、セキュア・プロセッサICカード315の場合は如何なるプログラム手段を講じても（ウイルス、クラッカーすべてを含め）鍵データそのものをICカードから外部へ取り出すこと、コピーすること、計測すること、その他観察行為すべてが難しく、不可能である。

#### 【図面の簡単な説明】

##### 【0031】

【図1】セキュア・プロセッサの構成例を示すブロック図である。

【図2】鍵データを読み出す部分のブロック図である。

【図3】セキュア・プロセッサの命令のフォーマット例を示す図である。

【図4-1】セキュア・プロセッサのオペコード表の一部分である。

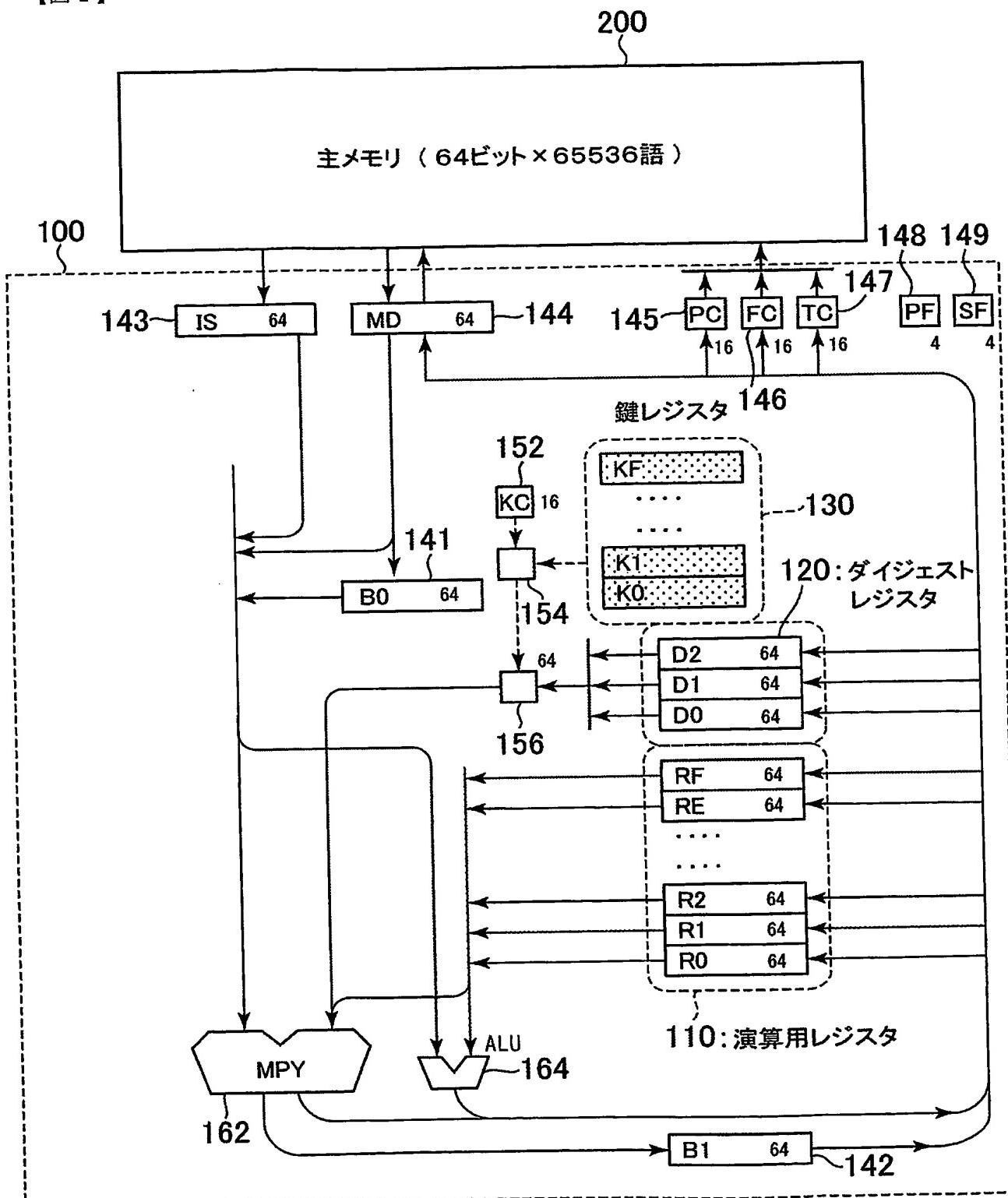
【図4-2】セキュア・プロセッサのオペコード表の続きと、オペコードのフィールド記号や動作の表記の説明の表等である。

【図5】署名演算を示すフローチャートである。

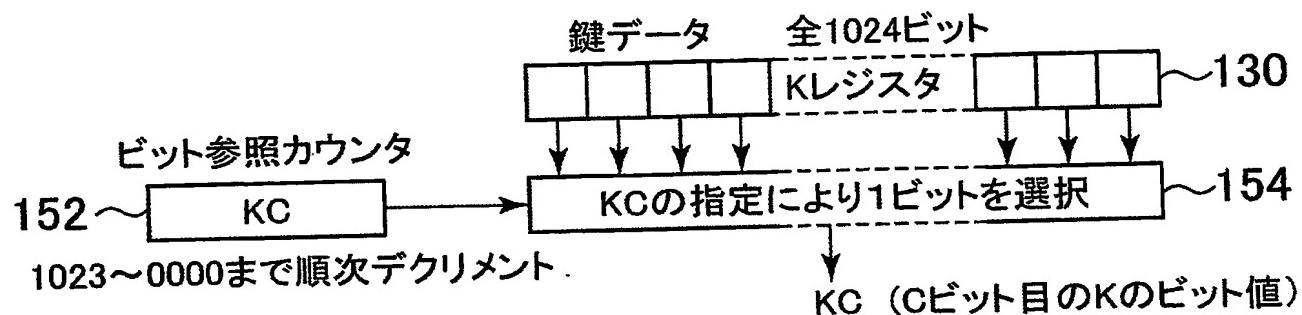
【図6】モントゴメリ演算をセキュア・プロセッサの命令で行う場合を示す図である。

【図7】セキュア・プロセッサをICカードに適用した構成例を説明する図である。

【書類名】 図面  
【図 1】



【図2】



【図3】

名稱	ビット長	内容
OP	4	Operation Code
SOP	4	Sub-Operation Code
MF	4	Mode F Operand
MT	4	Mode T Operand
L	16	Length
F	16	From Operand (F Operand)
T	16	To Operand (T Operand)
S	16	Sink Operand (S Operand)

(a)

オペランド指定モード	2進コード	モード内容
D	0000	「または」オペランドの値を演算用レジスタ番号と解釈し、そのレジスタ内容を参照する(直接レジスタ指定)
I	0001	「または」オペランドの値を演算用レジスタ番号と解釈し、そのレジスタ内容をアドレスとして、メモリを参照する(間接レジスタ指定)
A	0010	「または」オペランドの値をアドレスと解釈し、そのアドレスでメモリを参照する(直接アドレス指定)
P	0011	間接レジスタ指定をした後、参照したレジスタ値を+1する
M	0100	指定されたレジスタ値を-1し、その値をアドレスとしてメモリを参照する
V16	0101	オペランドに指定された16bit値を直接演算に使用する
V64	0110	次命令に指定された64bit値を直接演算に使用する
U	1000	間接レジスタ指定の倍長演算モード。「または」オペランドで指定されたレジスタ内容を倍長データの先頭アドレスとして、ルフィールドに指定された倍長のデータを演算する。
LA	1001	直接アドレスを倍長演算モード。「または」オペランドに指定されたアドレスを倍長データの先頭アドレスとして、ルフィールドに指定された倍長のデータを演算する。

(b)

【図4-1】

(a)

OP (4bit)	SOP (4bit)	MF (4bit)	MT (4bit)	L (16bit)	F (16bit)	T (16bit)	ニモニック	動作	PSW (N Z V C)	属性
0000		D	D				HLT	HLT		
0001 0000	D	-ttt			T	CLR		0 → T	0100	
0001 0001						CLRS			0100	
0010 0000	D	-ttt			T	ASL	T × 2 → T	**0*		SFTs
0010 0001	D	-ttt			T	ASR	T ÷ 2 → T	**0*		
0010 0010	D	tttt	L		T	LSL	T左論理シフト → T	**0*		
0010 0011	D	tttt	L		T	LSR	T右論理シフト → T	**0*		
0010 0100	D	tttt	L		T	LSLC	T左論理シフト(キャリーを含む) → T	**0*		
0010 0101	D	tttt	L		T	LSRC	T右論理シフト(キャリーを含む) → T	**0*		
0010 0110	D	-ttt			T	RSL	T左ローテート → T	**0*		
0010 0111	D	-ttt			T	RSR	T右ローテート → T	**0*		
0011 0000	ffff	tttt	L	F	T	ADD	T+F → T	****		ADDs
0011 0001	ffff	tttt	L	F	T	ADC	T + F + Cflag → T	****		
0011 0010	ffff	tttt	L	F	T	INC	T + 1 → T	****		
0011 0011	D	-ttt			T	NEG	-T + 1 → T	****		
0100 0000	ffff	tttt	L	F	T	SUB	T - F → T	****		SUBs
0100 0001	ffff	tttt	L	F	T	SBB	T - F - Cflag → T	****		
0100 0010	ffff	tttt	L	F	T	DEC	T - 1 → T	****		
0100 0011	ffff	tttt	L	F	T	CMP	T - F → T	****		
0101 0000	ffff	tttt	L	F	T	AND	T $\wedge$ F → T	**0-		BITs
0101 0001	ffff	tttt	L	F	T	OR	T $\vee$ F → T	**0-		
0101 0010	ffff	tttt	L	F	T	XOR	T $\oplus$ F → T	**0-		
0101 0011	ffff	tttt	L	F	T	NOT	-T → T	**0-		
0101 0100	-fff	-ttt		F	T	BIT	T $\wedge$ F → T	**0-		
0110 0000	ffff	tttt	L	F	T	MOV	F → T	**0-		MOVs
0110 0001	-fff	IP		F	SP	PUSH	F → (SP)+			
0110 0010	MI	-ttt		SP	T	POP	-(SP) → T			
0110 0011	-fff	D		F	?	IN	F → ?			
0110 0100	D	-ttt		F	?	OUT	? → T			
0111 0000	-fff	D		F	PC	JMP	F → PC			JMPs
0111 0001	-fff	D		F	PC	RJP	PC + F → PC			
0111 0010	MI	D		SP	PC	RET	-(SP) → PC			
0111 0011	MI	D		SP	PC	RIT	-(SP) → PC, ITF reset			
1000 0000	-fff	D		F	PC	JSR	PC → (SP)+, F → PC			LINKs
1000 0001	-fff	D		F	PC	RJS	PC → (SP)+, PC + F → PC			
1000 0010	-fff	D		F	PC	SVC	PC → (SP)+, F → PC, ITF set			
1001 0000	-fff	D		F	PC	BRN	[N=1] F → PC			BRs
1001 0001	-fff	D		F	PC	BRZ	[Z=1] F → PC			
1001 0010	-fff	D		F	PC	BRV	[V=1] F → PC			
1001 0011	-fff	D		F	PC	BRC	[C=1] F → PC			
1010 0000	-fff			F	PC	LOOP	(PC)-1 → (PC) [Z ≠ 1] F → PC	---		
1010 0001	ffff	D	0011	F		DMV	F(ダイジェスト) → (D0,D1,D2)			
1010 0010	-fff	tttt		F	T	XCHG	F → T, T → F			
1011 0000	-fff	-ttt		F	T	MUL	F × T → RF.RE	****		
1100 0000	D	D		SP	PC	SIG	PC → (SP)+, 固定番地 → PC SF set, KC初期化			LINKs
1100 0001	MI	D		SP	PC	SIE	[SF=1,KC=0] -(SP) → PC, SF reset			
1100 0010	-fff	D		F		KCJ	[SF=1-KCE ≠ 0] F → PC			
1100 0011	LA	LA	L	F	T	ADO	[SF=1] F + T → T			
1100 0100	LA	LA			T	SCMP	[SF=1] compare N with T			ROMs
1100 0101	LA	LA			T	SSB	[SF=1] T - N → T			

【図4-2】

(b)

OP (4bit)	SOP (4bit)	L (8bit)	F (16bit)	T (16bit)	S (16bit)	二モード	動作	PSW
1101 0000	L	F	T	S	MLS	[SF=1] F × T → S		
1101 0001	L		T	S	MDK	[SF=1] T × D <sup>Kc</sup> → S, KC-1 → KC		
1101 0010	L		T	S	MLD	[SF=1] T × D → S		
1101 0011	L		T	S	MUL	[SF=1] N'(rom) × T の下位 → S		MULs
1101 0100	L		T	S	MLH	[SF=1] N(rom) × T の上位 → S		
1101 0101	L		T	S	MLP	[SF=1] 定数 R <sup>2</sup> mod N(rom) × T → S		

(c)

フィールド	記号	説明	補足
D	D	Dモード固定,Dモードに対応する2進コードが入ります	*1,FまたはTオペランドを任意に指定出来ないにもかかわらず,モード指定がされない場合があります.これは制御上,レジスタやアドレスの指定が無くともその指定されたモードと同じ動作が必要であるためです.例:HLT,ASL等
I	I	IPモード固定,IPモードに対応する2進コードが入ります	
M	M	MIモード固定,MIモードに対応する2進コードが入ります	
M,F,M,T *,1,*2	LA	LAモード固定,LAモードに対応する2進コードが入ります	*2,モード指定の説明につきましては次シートの"モード説明をご覧下さい."
	U	Uモード固定,Uモードに対応する2進コードが入ります	*3,スタックポインタ(SP)はSEP-4ロック回には記されていませんが存在しています.
f		任意のビットを指定.	
t		任意のビットを指定.	
-		指定無し,仮に指定されても無視.	*4,間接参照はレジスタ内容をアドレスとしてメモリにアクセスし,そのアドレスに格納されている値を参照することです.
L		任意の倍長データの長さを指定	
L	0011	3倍長データ(64×3)を固定で指定	
F,T	F	レジスタ番号またはアドレスデータの指定. モードによつて意味が異なります.*2	
PC,*3	T	レジスタ番号またはアドレスデータの指定. モードによつて意味が異なります.*2	
SP	PC	プログラムカウンタ(PC)レジスタを指定	
?	SP	スタックポインタ(SP)レジスタを指定	
S	?	未設計. 指定先を未決定.	
S	S	主メモリの上位の指定アドレスを指定	
*	*	Don't care(まだ方がります)	
PSW	0	使用されません	*5,"SF=1"の条件が付く命令においてJIF,LT,Sオペランドは署名計算で使用される特定のアドレスのみしか使用されません.
	1	固定で0	
		固定で1	

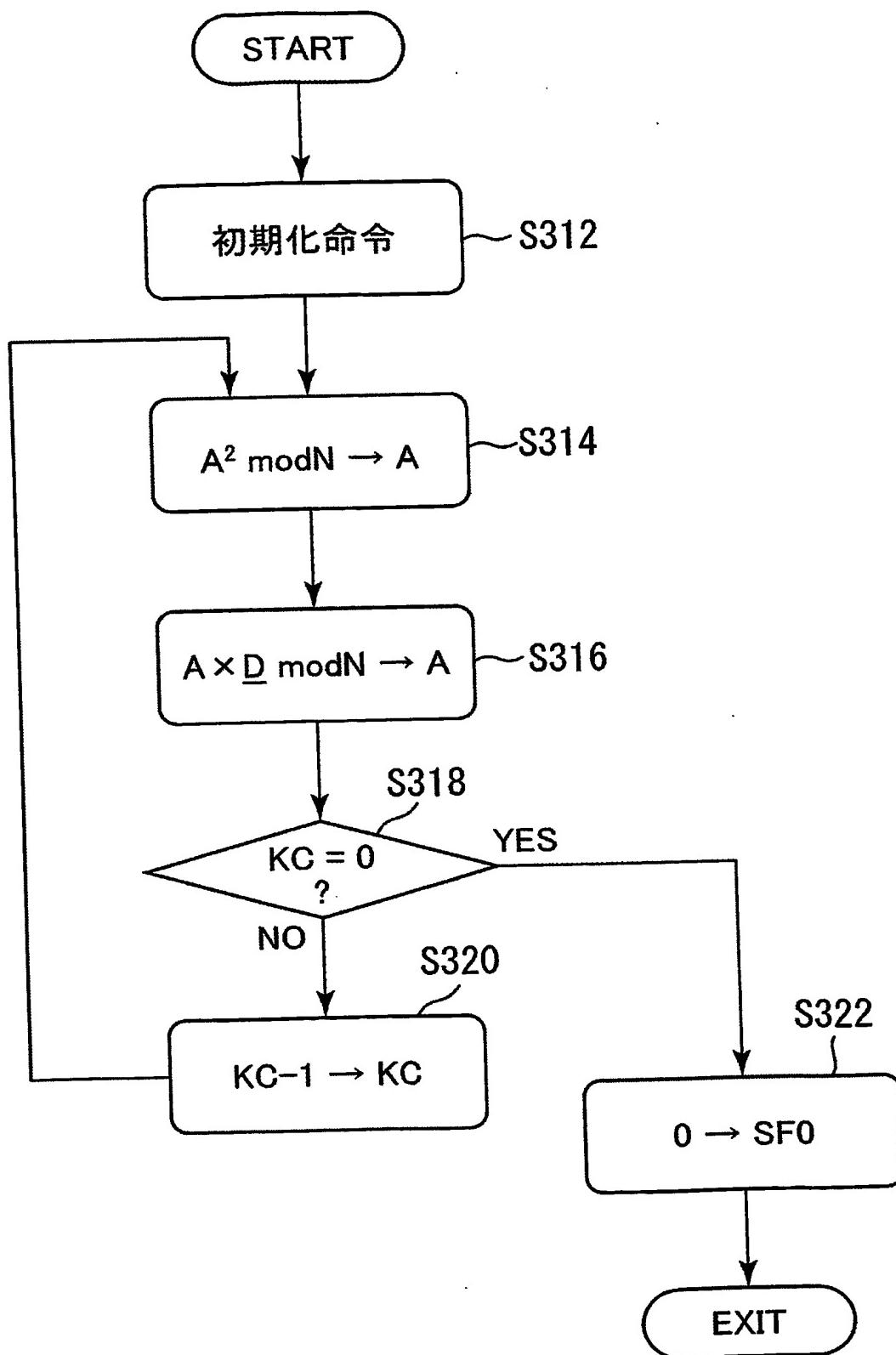
(d)

動作の表記について		意味
記号	V	AND 演算
	Λ	OR 演算
	▽	XOR 演算
	-	NOT 演算
	(~)	～の値を間接参照し～[c+4]
	(~)+	～の値を間接参照し～[c+1]
	(~)-	～を-1しその値を間接参照
	[~]	～を条件とする。

(e)

動作の表記について		意味
記号	V	AND 演算
	Λ	OR 演算
	▽	XOR 演算
	-	NOT 演算
	(~)	～の値を間接参照し～[c+4]
	(~)+	～の値を間接参照し～[c+1]
	(~)-	～を-1しその値を間接参照
	[~]	～を条件とする。

【図5】



【図6】

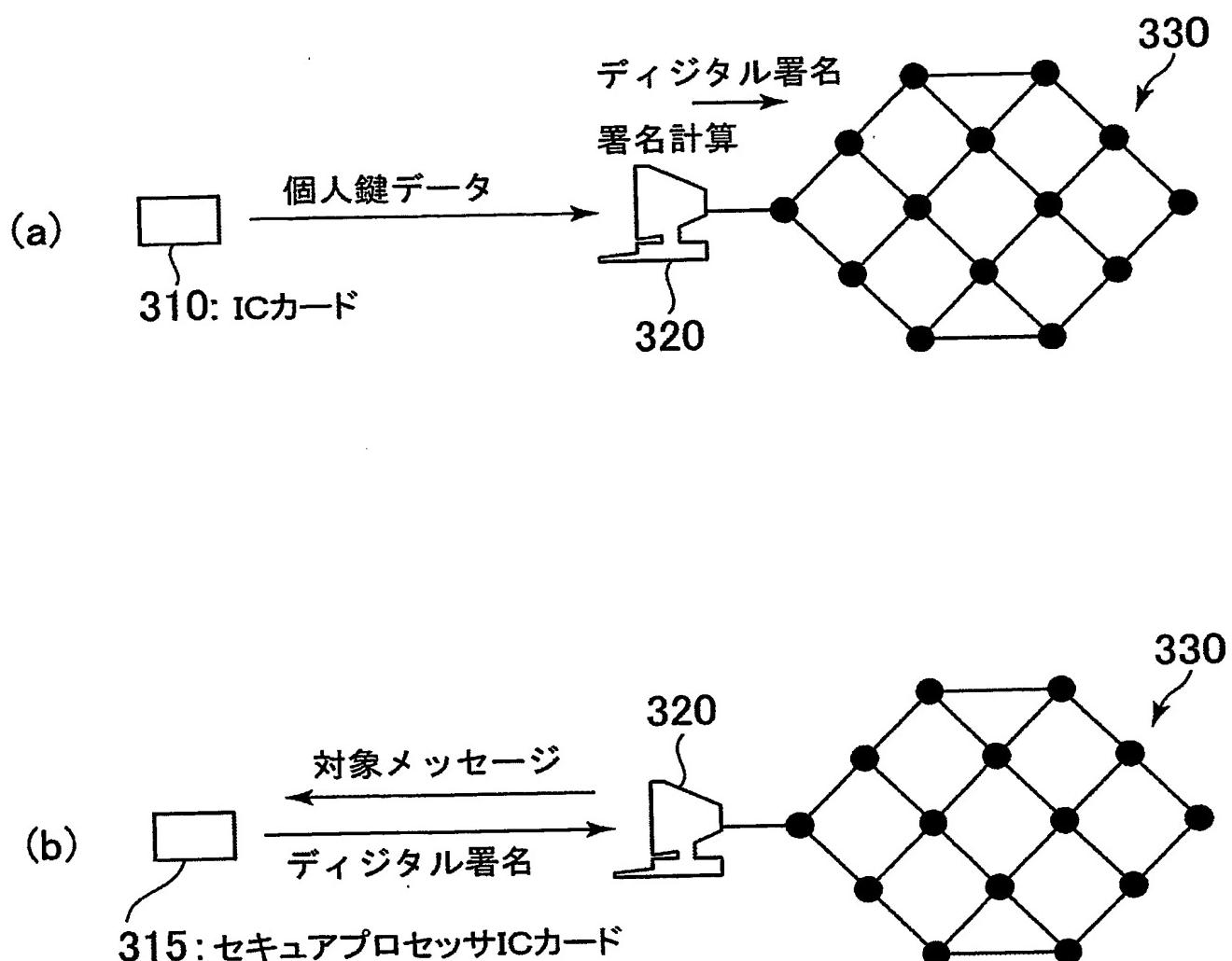
		動作		補足説明	
手順	命令				
(01)	AR*	MLP	R* × A → Z	Zの上位1024bitをZH、下位1024bitをZLとする特別、計算の必要はない。	<p>"XYR<sup>-1</sup>modN"という関数に対し、X=R<sup>2</sup> mod N, Y=Aを入力する。"XYR<sup>-1</sup>modN"の実際には"(04)×N"の上位で下位は無視できる。ADOで(06)と(07)の処理を一括できることから"ZH+AC+1"は必ずRの倍数となるからである。</p> <p>比較結果は次の命令に反映される。比較結果からNを引くか判断される。その結果ACは"AR<sup>2</sup>modN"となる。</p> <p>Zの上位1024bitをZH、下位1024bitをZLとする特別、計算の必要はない。</p> <p>Zの上位1024bitをZH、下位1024bitをZLで一度に(11)と(12)を処理する。</p> <p>(14)で実際には"(04)×N"の上位で下位は無視できる。ADOで(14)と(15)の処理を一括できることから"ZH+AC+1"は必ずRの倍数となるからである。</p> <p>比較結果は次の命令に反映される。比較結果からNを引くか判断される。その結果ACは"AD mod N"となる。</p>
(02)	AR* mod R				
(03)	(02) × N*	MLL	"N* × ZL"の下位1024bit → U	MLLで一度に(03)と(04)を処理する。	
(04)	(03) mod R				
(05)	(04) × N	MLH	"N × U"の上位1024bit → AC	(06)で実際には"(04)×N"の上位で下位は無視できる。	
(06)	(01)+(05)	ADO	ZH + AC + 1 → AC	ADOで(06)と(07)の処理を一括できることから"ZH+AC+1"は必ずRの倍数となる。	
(07)	(06) / R	SCMP	AC と N を比較	比較結果は次の命令に反映される。	
(08)	(07) - N	SSB	[AC > N] AC - N → AC	比較結果からNを引くか判断される。その結果ACは"AR <sup>2</sup> modN"となる。	
(09)	(08) × D	MDK	AC × D <sup>le</sup> → Z	Zの上位1024bitをZH、下位1024bitをZLとする特別、計算の必要はない。	
(10)	(09) mod R				
(11)	(10) × N*	MLL	"N* × ZL"の下位1024bit → U	MLLで一度に(11)と(12)を処理する。	
(12)	(11) mod R				
(13)	(12) × N	MLH	"N × U"の上位1024bit → AC	(14)で実際には"(04)×N"の上位で下位は無視できる。	
(14)	(09)+(13)	ADO	ZH + AC + 1 → AC	ADOで(14)と(15)の処理を一括できることから"ZH+AC+1"は必ずRの倍数となるからである。	
(15)	(06) / R	SCMP	AC と N を比較	比較結果は次の命令に反映される。	
(16)	(14) - N	SSB	[AC > N] AC - N → AC	比較結果からNを引くか判断される。その結果ACは"AD mod N"となる。	

( a )

記号の意味	
R*	定数: R <sup>2</sup> mod N
R	定数: R
N	定数: N
N*	定数: NN* mod R = R - 1 を満たす値
A	任意の値
D	ダイジエスト
Z	一時的な変数 2048bit
ZH	Zの上位1024bit
ZL	Zの下位1024bit
U	一時的な変数 1024bit
AC	累積される中間結果 1024bit

( b )

【図7】



**【書類名】要約書****【要約】**

**【課題】**汎用機能を持ち、なおかつ、セキュリティ機能（すなわち鍵データの安全保管とデジタル署名計算の高速化）も持つプロセッサの提供。

**【解決手段】**一般用と署名演算用の命令を有するセキュア・プロセッサ100内に、不揮発性の鍵レジスタ130に、鍵データが格納されている。鍵ビット参照カウンタ152は1023から0まで、順次デクリメントする。この鍵ビット参照カウンタ152の内容により、ビット指定ゲート154から、鍵レジスタ130に格納されているKデータが順次1ビットづつ指定され、逐次署名演算に使用される。鍵Kを格納する鍵レジスタ130から他へデータを転送する語データ並列伝送経路を設けていない。

このようなハードウェア構造からして、鍵データを生の形で直接外部へ出すことは不可能である。

**【選択図】**図1

## 認定・付加情報

特許出願の番号	特願2003-380114
受付番号	50301856428
書類名	特許願
担当官	末武 実 1912
作成日	平成15年11月13日

## &lt;認定情報・付加情報&gt;

【提出日】	平成15年11月10日
-------	-------------

特願 2003-380114

## 出願人履歴情報

識別番号 [503360115]

1. 変更年月日 2003年10月 1日

[変更理由] 新規登録

住所 埼玉県川口市本町4丁目1番8号  
氏名 独立行政法人 科学技術振興機構

2. 変更年月日 2004年 4月 1日

[変更理由] 名称変更

住所 埼玉県川口市本町4丁目1番8号  
氏名 独立行政法人科学技術振興機構

# **Document made available under the Patent Cooperation Treaty (PCT)**

International application number: PCT/JP04/016589

International filing date: 09 November 2004 (09.11.2004)

Document type: Certified copy of priority document

Document details: Country/Office: JP  
Number: 2003-380114  
Filing date: 10 November 2003 (10.11.2003)

Date of receipt at the International Bureau: 27 January 2005 (27.01.2005)

Remark: Priority document submitted or transmitted to the International Bureau in compliance with Rule 17.1(a) or (b)



World Intellectual Property Organization (WIPO) - Geneva, Switzerland  
Organisation Mondiale de la Propriété Intellectuelle (OMPI) - Genève, Suisse